A 11.5-Gbps LDPC Decoder Based on CP-PEG Code Construction

Chih-Lung Chen, Kao-Shou Lin, Hsie-Chia Chang, Wai-Chi Fang, and Chen-Yi Lee

Department of Electronics Engineering, National Chiao Tung University,

1001 Ta-Hsueh Road, Hsinchu, Taiwan, R.O.C.

Tel: +886-3-5131369 email: {lung, hcchang}@si2lab.org

Abstract—In this paper, a LDPC decoder chip based on CP-PEG code construction is presented. The (2048, 1920) irregular LDPC code generated by CP-PEG algorithm has better performance than other PEG-based codes; however, the large check node degrees introduced by high code-rate 15/16 become the implementation bottleneck. To design such a high code-rate LDPC decoder, our approach features *variable-node-centric sequential scheduling* to reduce iteration number, single pipelined decoder architecture to lessen the message storage memory size, as well as optimized check node unit to further compress the register number. Overall 73% message storage memory is saved as compared with traditional architecture. Fabricated in 90nm 1P9M CMOS technology, a test decoder chip could achieve maximum 11.5 Gbps throughput under 1.4V supply voltage with core area of $2.7 \times 1.4 \text{ mm}^2$. The energy efficiency is only 0.033 nJ/bit with 5.77 Gbps at 0.8V to meet IEEE 802.15.3c requirements.

Index Terms—LDPC, High throughput, sequential scheduling.

I. INTRODUCTION

Low-density parity-check (LDPC) code is a famous error control code with near Shannon limit performance [1] and can be described by its parity-check matrix H . The rows and columns of H are mapped to check nodes and variable nodes of a bipartite graph, on which the belief-propagation (BP) algorithm exchanges messages between nodes iteratively to decode LDPC codes [2]. The message exchanging order between nodes is called scheduling, which will influence the convergence speed of the decoding algorithm. In standard BP algorithm, simultaneous update of all check node messages or variable node messages is named as *flooding scheduling*. Alternatively, the layered BP algorithm [3] performing message update along different check node groups is a method of *check-node-centric sequential scheduling* (CSS). Researches have revealed that CSS could reduce maximum iteration to approximate half of the standard BP with similar performance.

Recently, LDPC codes adopted in high-throughput systems have high code-rate property to increase channel efficiency. However, the introduced large check node degree d_c will cause implementation full of difficulties. For example, the largest check node degree of (2048, 1723) LDPC code adopted in IEEE 802.3an [4] equals 32, leading to routing congestion and low chip density. Even though the CSS could reduce the iteration number, the throughput is still degraded due to long critical path of check node unit (CNU). The situation will become worse for the (1440, 1344) LDPC code of IEEE 802.15.3c [5] with $d_c = 45$.

This work was supported by the NSC, Taiwan, R.O.C., under grant NSC 97-2220-E-009-013. The authors would like to thank UMC, MediaTek, CIC, and Mr. Yi-Kai Lin for their assistance.

In this paper, the proposed decoder aims at providing a high-throughput and hardware-efficient solution to the high code-rate LDPC with large check node degrees. In order to reduce the iteration number, the decoding scheduling is based on the *variable-node-centric sequential scheduling* (VSS; also known as shuffled decoding [6]), where the messages are updated along different variable node groups. Since the inputs of CNU operation are also divided into several subgroups, the complexity and critical path delay of CNU are reduced. Furthermore, single pipelined approach and modified CNU are proposed to minimize the message storage memory. Using a (2048, 1920) LDPC code constructed by circulant permutation progressive edge-growth (CP-PEG) algorithm [7] as a design example, the overall decoder chip implemented in 90nm technology will show its advantages in terms of throughput, energy efficiency, and hardware efficiency.

The rest of this paper is organized as follows. Section II introduces the code structure and the decoding algorithm with VSS. In Section III, we propose a modified scheduling algorithm and an improved decoder architecture. Performance simulation and implementation result are shown in Section IV. Finally, the conclusion is given in Section V.

II. CODE STRUCTURE AND DECODING ALGORITHM

A. CP-PEG LDPC Code Construction

The (2048, 1920) irregular LDPC code, rate-15/16, used in this paper was constructed by CP-PEG algorithm and shown in Fig. 1(a). The constructed parity-check matrix H consists of $p \times p$ circulant permutation (CP) and all-zero matrices. A CP matrix is a cyclic square matrix with constant row and column weight of one. The number of each CP matrix indicates the cyclic shift amount and -1 means all zero matrix. By setting $p = 32$, there are $4 \times p$ check nodes and $64 \times p$ variable nodes in bipartite graph, where each check node has uniform degree 46, and $16 \times p$, $24 \times p$, $24 \times p$ variable nodes have degrees of 4, 3, 2 respectively. The performance of this code was proven to have better performance than other PEG-based LDPC codes [7]; nevertheless, the high check node degree required suitable decoder architecture to overcome implementation difficulties.

B. Variable-node-centric Sequential Scheduling

In VSS approach, the initialization, stopping criterion test, and output steps remain the same as the standard BP algorithm. The only difference between two algorithms lies in the updating procedure. The normalized min-sum (NMS) algorithm which compensates the approximation error in check node

Fig. 1. Parity-check matrix of (2048, 1920) LDPC code.

update step can also be applied to our VSS approach with normalized factor $\beta = 0.75$.

Assume the N bits of a codeword are divided into G groups, so each group contains $N/G \equiv N_G$ bits. The log-likelihood ratio (LLR) of intrinsic information of the n -th variable node is denoted by P_n . The message from n -th variable node to m -th check node is denoted by z_{mn} . The message from m -th check node to n -th variable node is denoted by ε_{mn} . The *a posteriori* LLR of n -th bit is denoted by z_n . The updating procedure of VSS approach with NMS algorithm is carried out as follows:

- 1) **Initialization:** $z_{mn}^{(0)} = P_n$,
- 2) **Iterative Decoding:** For $0 \leq g \leq G-1$, perform the following two steps.

- a) Check node to variable node update step, for $g \cdot N_G \leq n \leq (g+1) \cdot N_G - 1$ and each $m \in M(n)$, process

$$\varepsilon_{mn}^{(i)} \approx \prod_{\substack{n' \in N(m) \setminus n \\ n' \leq g \cdot N_G - 1}} \text{sign}(z_{mn'}^{(i)}) \times \prod_{\substack{n' \in N(m) \setminus n \\ n' \geq g \cdot N_G}} \text{sign}(z_{mn'}^{(i-1)}) \times \min \left\{ \min_{\substack{n' \in N(m) \setminus n \\ n' \leq g \cdot N_G - 1}} \left\{ |z_{mn'}^{(i)}| \right\}, \min_{\substack{n' \in N(m) \setminus n \\ n' \geq g \cdot N_G}} \left\{ |z_{mn'}^{(i-1)}| \right\} \right\} \times \beta \quad (1)$$

- b) variable node to check node update step, for $g \cdot N_G \leq n \leq (g+1) \cdot N_G - 1$, process

$$z_{mn}^{(i)} = P_n + \sum_{m' \in M(n) \setminus m} \varepsilon_{m'n}^{(i-1)} \quad (2)$$

$$z_n^{(i)} = P_n + \sum_{m \in M(n)} \varepsilon_{mn}^{(i-1)} \quad (3)$$

- 3) **Hard Decision:** Let X_n be the n -th bit of decoded codeword. If $z_n^{(i)} \geq 0$, $X_n = 0$, else if $z_n^{(i)} < 0$, $X_n = 1$.

In this work, the codeword is divided into $G = 4$ groups, therefore the parity-check matrix H is divided into 4 submatrices (H1 to H4). As shown in Fig. 1(b), each submatrix consists

of equal number of variable nodes with the same degree to reduce the hardware cost of variable node unit (VNU). Moreover, the submatrices with the same shift amounts (shaded blue CP matrices) are arranged in the same position thus the routing and control could be further simplified.

III. PROPOSED DECODER ARCHITECTURE

In this section, a complete decoder architecture will be presented, including datapath, scheduling, and VLSI structure of CNU and modified CNU.

A. Single Pipelined Architecture

The entire decoder depicted in Fig. 2(a) is composed of fully-parallel CNUs and partial-parallel VNUs, where the VNU2, VNU3, and VNU4 will handle variable node operations with degree 2, 3, and 4 respectively. Let $\alpha_g^{(i)}$ denote the sorted messages sent from variable nodes in the g -th group to one specific check node at i -th iteration, which is:

$$\alpha_g^{(i)} = \min_{\substack{n' \in N(m) \setminus n \\ g \cdot N_G \leq n' \leq (g+1) \cdot N_G - 1}} \left\{ |z_{mn'}^{(i)}| \right\} \quad (4)$$

Then the magnitude part of check node to variable node message in (1) could be computed by the following equation:

$$| \varepsilon_{mn}^{(i)} | = \min \left\{ \left\{ \alpha_j^{(i)} \right\}_{j < g}, \alpha_g^{(i)}, \left\{ \alpha_k^{(i-1)} \right\}_{k > g} \right\} \quad (5)$$

Fig. 2(b) demonstrates the timing diagram of proposed decoder. There are G initialization cycles required to calculate α_g^0 for $0 \leq g \leq G-1$. Since only one subgroup of the message z_{mn} is updated in g -th cycle of one iteration, the main operation of CNU could be simplified to calculate $\alpha_g^{(i)}$ (local sorting) in each cycle and then perform global sorting like equation (5).

From the proposed single pipelined architecture, only messages $\alpha_g^{(i)}$ and $\varepsilon_{mn}^{(i)}$ are stored. The sorted results could be represented by min value, second min value, and the index of min value in NMS algorithm. Therefore, the proposed decoder only latches two values, one index, and sign part of messages in each subgroup, while the variable node to check node message $z_{mn}^{(i)}$ is on-the-fly calculated. The single pipelined architecture is feasible because the CNU could be updated immediately after VNU's operations in VSS approach.

B. Modified CNU

The operation of check node to variable node update could be divided into magnitude part and sign part. Fig. 3(a) illustrates the magnitude part of CNU, which is an accumulative sorter composed of a local sorter and a global sorter. The local sorter is used to find the local min and second min values in each subgroups, and global min and second min values of a check node will be found by a global sorter. Similarly, the sign operation can be computed in an accumulative way like the accumulative sorter.

For our proposed CP-PEG LDPC codes with $dc = 46$, The VSS approach with $G = 4$ could divide 46 inputs of the sorter into only 12 inputs. More subgroup number G will result in

(a) Single pipelined LDPC decoder architecture

(b) variable-node-centric sequential scheduling

Fig. 2. Proposed architecture and scheduling.

fewer inputs of sorter, but increase the storage for min, second min, and index values of each subgroup.

In order to further reduce the storage overhead of each subgroup, we propose a reduced storage accumulative sorter as shown in Fig.3(b). The basic idea is to simplify the local min and local second min from $G - 1$ subgroups into one group. Some extra control circuits are needed to open or close the feedback loop in Fig.3(b). This sorter architecture is beneficial since the complexity reduction of storage registers and global sorters is higher than the overhead of control circuits. Section IV will show the performance of this modified CNU is similar to original CNU.

C. Summary

In traditional two-stage pipelined architecture, both $z_{mn}^{(i)}$ and $\varepsilon_{mn}^{(i)}$ messages are kept in registers or memory. Assume the bit-width of messages is w ($= 6$) and variable node degree is dv , then the required memory size (or registers) is as follows:

$$\begin{aligned}
 & MEM_{VNU} + MEM_{CNU} \\
 &= \left(\sum dv \right) \cdot w + \\
 & \quad (N - K) \cdot (Min + 2ndMin + Index + Sign) \\
 &= \left(\sum dv \right) \cdot w + (N - K) \cdot (2 \cdot (w - 1) + \log_2[dc] + dc) \\
 &= 5888 \cdot 6 + 128 \cdot (2 \cdot 5 + \log_2[46] + 46) = 43264 \quad (6)
 \end{aligned}$$

For the proposed single pipelined decoder and modified

(a) Accumulative sorter

(b) Reduced storage accumulative sorter

Fig. 3. CNU architecture.

CNU in Fig.3(b), the memory size is reduced to

$$\begin{aligned}
 & MEM_{CNU} \\
 &= (N - K) \cdot \\
 & \quad (2 \cdot (Min + 2ndMin + Index + 2ndIndex) + Sign) \\
 &= (N - K) \cdot (4 \cdot (w - 1) + 4 \cdot \log_2[dc] + dc) \\
 &= 128 \cdot (4 \cdot 5 + 4 \cdot \log_2[46] + 46) = 11520 \quad (7)
 \end{aligned}$$

Therefore the overall register reduction of proposed architecture is 73%, leading to the following advantages: fewer registers, higher utilization of functional units, and reduced complexity. Since high-rate LDPC codes usually have more VNUs than CNUs (in our case: 512 VNUs and 128 CNUs), the elimination of registers from VNU to CNU not only reduces hardware cost but also lowers power consumption of clock tree.

IV. PERFORMANCE AND IMPLEMENTATION RESULTS

Under AWGN channel with BPSK modulation, the performance curves are simulated to determine the required bit-width and maximum iteration number. The simulation parameters of proposed algorithm are 6-bit input quantization (5-bit integer and 1-bit decimal fraction), scaling factor 0.75 for NMS algorithm, and 4 or 5 iterations. In Fig. 4, the bit-error rate (BER) curves indicate that 4 iterations for the proposed algorithm are sufficient to achieve similar performance of standard BP algorithm with 7 iterations. Furthermore, in the aspect of almost the same code-rate and better error-correcting capability, our CP-PEG LDPC codes outperforms 1.2 dB better than the (255, 239) RS code at $BER=10^{-5}$, which reveals the potential of CP-PEG LDPC codes for storage applications and fiber optical communication systems. The overall SNR loss between this work and Shannon limit is only 1.6dB.

The proposed LDPC decoder is implemented by standard-cell design flow and fabricated in 90-nm 1P9M CMOS technology. The core occupied 3.84 mm^2 area with 68%

Fig. 4. Performance.

utilization. The die photo is shown in Fig. 5, where the distribution of CNU and VNU is auto-determined by APR tool. Since required decoding cycles of one LDPC codeword are 4 initialization cycles plus 4 iterations, the throughput is $(1920\text{bit}/20\text{cycles}) \times \text{frequency}$. Fig. 6 shows the measured maximum throughput and power dissipation under different SNR conditions and supply voltages. The measurement result indicates that the test chip with FF corner can achieve 11.5 Gbps throughput under 1.4V supply voltage. The throughput could be scaled down to 5.77Gbps with 0.8V supply voltage to meet the throughput requirement of IEEE 802.15.3c standard and the energy efficiency will be 0.033 nJ/bit.

Compared with the state-of-the-art in Table I, the proposed LDPC decoder outperforms others in the aspects of throughput, hardware efficiency, and power efficiency. Since the LDPC code specification of these designs are different, the SNR loss between each work to their Shannon limit is also listed for reference.

V. CONCLUSION

A high-throughput and power-efficient LDPC decoder is presented. Utilizing the characteristic of variable-node-centric sequential scheduling, the proposed decoding algorithm could reduce the maximum iteration without performance loss. In addition, the single pipelined architecture and modified CNU can save 73% message storage memory and decrease the sorter size, resulting in a low-complexity design. After implementation in 90nm technology, the test chip occupies 3.84 mm^2 area and supports maximum 11.5 Gbps data rate under 1.4V supply voltage.

Fig. 5. Chip micrograph.

Fig. 6. Measured maximum throughput and power consumption.

TABLE I
COMPARISON WITH STATE-OF-THE-ART

	This work	CICC'07 [8]	SOVC'07 [9]
Process	90-nm	0.13- μm	0.13- μm
Code Spec	(2048, 1920)	(660,480)	Wimax
Code Rate	0.9375	0.73	0.5
Core Area	3.84 mm^2	7.3 mm^2	4.45 mm^2
Gate Count	708k	690k	420k
Iteration	4	15	2 - 8
Input Quantization	6 bits	4 bits	8 bits
Clock Frequency	120 MHz	300 MHz	83.3 MHz
Max. Throughput	11.5 Gbps	2.44 Gbps	222 Mbps
Power	191.2 mW ¹	1383 mW ²	52 mW
Energy Efficiency	0.033 nJ/bit ¹	0.566 nJ/bit ²	0.23 nJ/bit
Hardware Efficiency ³	12.1	3.5	0.53
SNR loss to Shannon limit ⁴	1.6 dB	2.8 dB	2.9 dB

¹ when frequency is scaled to 5.77 Gbps throughput with 0.8V supply voltage where BER=10⁻⁶

² when $E_b/N_0=5.5\text{dB}$ indicating BER < 10⁻⁸

³ Throughput/Gate count (Mbps/K-gate)

⁴ when BER=10⁻⁵

REFERENCES

- [1] R. G. Gallager, *Low-Density Parity-Check Codes*. Cambridge, MA: MIT Press, 1963.
- [2] D. J. C. MacKay and R. M. Neal, "Near Shannon limit performance of low density parity check codes," *Electron. Lett.*, vol. 33, no. 6, pp. 457–458, Mar. 1997.
- [3] M. Mansour and N. Shanbhag, "High-throughput LDPC decoders," *IEEE Trans. on VLSI Systems*, vol. 11, no. 6, pp. 976–996, Dec. 2003.
- [4] *Part 3: carrier sense multiple access with collision detection (CSMA/CD) access method and physical layer specifications*, IEEE Std. P802.3an-2006, Sept. 2006.
- [5] *Part 15.3: wireless medium access control (MAC) and physical layer (PHY) specifications for high rate wireless personal area networks (WPANs)*, IEEE Std. P802.15.3c-DF8, 2009.
- [6] J. Zhang and M. Fossorier, "Shuffled iterative decoding," *IEEE Transactions on Communications*, vol. 53, no. 2, pp. 209–213, Feb. 2005.
- [7] Y. K. Lin, C. L. Chen, Y. C. Liao, and H. C. Chang, "Structured LDPC codes with low error floor based on peg tanner graphs," in *IEEE Int. Sympo. Circuits and Systems (ISCAS'08)*, May 2008, pp. 1846–1849.
- [8] A. Darabiha, A. C. Carusone, and F. R. Kschischang, "A 3.3-Gbps bit-serial block-interlaced min-sum LDPC decoder in 0.13- μm CMOS," in *Proc. IEEE CICC'07*, Sept. 2007, pp. 459–462.
- [9] X. Y. Shih, C. Z. Zhan, C. H. Lin, and A. Y. Wu, "A 19-mode 8.29mm² 52-mW LDPC decoder chip for IEEE 802.16e system," in *Proc. Int. Sympo. VLSI Circuits (SOVC'07)*, June 2007, pp. 16–17.